



IEEE Standard for Automatic Test Markup Language (ATML) for Exchanging Automatic Test Equipment and Test Information via XML

IEEE Standards Coordinating Committee 20

Sponsored by the
IEEE Standards Coordinating Committee 20 on Test and Diagnosis for
Electric Systems

IEEE
3 Park Avenue
New York, NY 10016-5997, USA

14 February 2009

IEEE Std 1671™-2006

IEEE Standard for Automatic Test Markup Language (ATML) for Exchanging Automatic Test Equipment and Test Information via XML

Sponsor

IEEE Standards Coordinating Committee 20 on Test and Diagnosis for Electronic Systems (SCC20)

Approved 15 September 2006

IEEE-SA Standards Board

Abstract: This standard specifies the framework for the family of ATML standards. ATML defines a standard exchange medium for sharing information between components of an Automatic Test System (ATS), using the eXtensible Markup Language (XML).

Keywords: Automatic Test Equipment (ATE), Automatic Test Markup Language (ATML), ATML instance document, automatic test system (ATS), XML Schema

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Copyright © 2009 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 14 February 2009. Printed in the United States of America.

IVI is a registered trademark in the U.S. Patent & Trademark Office, owned by the IVI Foundation, Inc.

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by the Institute of Electrical and Electronics Engineers, Incorporated.

PDF: ISBN 978-0-7381-5901-0 STD95904
Print: ISBN 978-0-7381-5902-7 STDPD95904

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

Use of an IEEE Standard is wholly voluntary. The IEEE disclaims liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEEE Standard document.

The IEEE does not warrant or represent the accuracy or content of the material contained herein, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEEE Standards documents are supplied “**AS IS**.”

The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

In publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon his or her independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration. A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal interpretation of the IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position, explanation, or interpretation of the IEEE.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Comments on standards and requests for interpretations should be submitted to the following address:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854
USA

Authorization to photocopy portions of any individual standard for internal or personal use is granted by The Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Introduction

This introduction is not part of IEEE Std 1671-2006, IEEE Standard for Automatic Test Markup Language (ATML) for Exchanging Automatic Test Equipment and Test Information via XML.

The benefits of using standards in test-related applications have long been recognized. The scope for standardization extends from low-level standards associated with test instrument control to high-level standards associated with specifying tests in an implementation-independent manner. In the 1960s, Aeronautical Radio, Incorporated (ARINC) started the development of the Abbreviated Test Language for Avionics Systems. In 1976, management of the ATLAS Standard was passed to the IEEE, and the ATLAS acronym was changed to Abbreviated Test Language for All Systems to reflect its broader field of applications.

Within the IEEE, development of ATLAS and ATLAS-related standards was vested in an ad hoc committee, which later became the IEEE Standards Coordinating Committee 20 (SCC20). In the mid-1980s, SCC20 broadened the scope of its activities to embrace other standards projects related to test and diagnosis, including Automatic Test Program Generation (ATPG), Test Equipment Description Language (TEDL), Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE), A Broad Based Environment for Test (ABBET), Software Interface to Maintenance Information and Collection Analysis (SIMICA), Receiver/Fixture Interface (RFI), Signal and Test Definition (STD), and Automatic Test Markup Language (ATML).

This standard provides the framework for a family of standards providing specifications for test-related applications and environments. This family specifies language-independent elements within a wide variety of test environments, including built-in test (BIT) systems, automatic test systems (ATS), and manual test environments. Each of these interfaces is specified in the form of a XML schema.

XML schemas define the basic information required within any test application and provide a vehicle for formally defining the test environment by defining a class hierarchy corresponding to these basic information entities and provide several methods within each to enable basic operations to be performed on these entities. ATML components standards, within the ATML framework, define the particular requirements within the test environment.

Notice to users

Laws and regulations

Users of these documents should consult all applicable laws and regulations. Compliance with the provisions of this standard does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

This document is copyrighted by the IEEE. It is made available for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making this document available for use and adoption by public authorities and private users, the IEEE does not waive any rights in copyright to this document.

Updating of IEEE documents

Users of IEEE standards should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect. In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE Standards Association web site at <http://ieeexplore.ieee.org/xpl/standards.jsp>, or contact the IEEE at the address listed previously.

For more information about the IEEE Standards Association or the IEEE standards development process, visit the IEEE-SA web site at <http://standards.ieee.org>.

Errata

Errata, if any, for this and all other standards can be accessed at the following URL: <http://standards.ieee.org/reading/ieee/updates/errata/index.html>. Users are encouraged to check this URL for errata periodically.

Interpretations

Current interpretations can be accessed at the following URL: <http://standards.ieee.org/reading/ieee/interp/index.html>.

Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this standard was completed, the Test Information Integration (TII) Subcommittee had the following membership:

Chris Gorringer, *Co-Chair*

Mike Seavey, *Co-Chair*

Hakan Akgun
Anthony Alwardt
Jack Amsell
David Barton
Keith Beard
Michael Bodkin
Malcolm Brown
Antonius Bunsen
Darryl Busch
Carl Byington
Giampiero Casalegno
Steve Cmiel
Kevin Coggins
Matt Cornish
Bernard Dathy
Timothy Davis
David Droste
James Dumser
Tamara Einspanjer
William Eklow
Keith Ellis
Leo Errico
Oscar Fandino
Jennifer Fetherman
Ken Finklea
Melissa Ford
Robert Fox
Brit Frank
Thomas Gaudette
Thomas Gauntner
Scott Gearhart
George Geathers

Anthony Geneva
William E. Gerstein
Sudipto Ghoshal
Gerald Goldemund
Arnold Greenspan
Kyle Gupton
Arthur Hann
Michelle Harris
Michael Harrison
Robert Hayes
Alicia Helton
Hans Hopf
Ashley M. Hulme
Ivor Isaacs
Anand Jain
Patrick Kalgren
John Knowles
Guy Larcom
Phillip Liang
Jay Lindsey
Teresa P. Lopes
Kenneth Lynch
Michael Malesich
David Mills
Scott Misha
Mukund U. Modi
Ion A. Neag
Angela Nielson
Steven O'Donnell
Leslie A. Orlidge
Stephen A. Osella
Dan Pleasant

Hugh Pritchett
John E. Ralph
Narayanan Ramachandran
Rabindranath Raul
Peter Richardson
Robert A. Robinson
David A. Rohacek
John Rosenwald
William Ross
Paul Salopek
Edward Sanchez
Thomas Sarfi
Howard Savage
David Sharone
John W. Sheppard
Henry Silcock
Roger J. Sowada
Joseph J. Stanco
Michael Stora
Walter Struppler
Ronald G. Taylor
Siow-khiang Tech
Hael Thibeault
David Thomas
Kirk Thompson
Vincent Tume
Joerg Urban
Brian Vestyck
David Walihermfechtcl
Timothy Wilmering
Vernon Woodward

The following members of the balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Ali Al Awazi
William A. Byrd
Danila Chernetsov
Keith Chow
Ryon K. Coleman
Tommy P. Cooper
Thomas J. Dineen
James Dumser
Heiko Ehrenberg
Tamara Einspanjer
Andrew C. Fieldsend
William B. Frank
Ignacio Marin Garcia
Thomas Gaudette
William E. Gerstein
Chris C. Gorringer
Randall C. Groves
Werner Hoelzl
Dennis Horwitz
Ashley M. Hulme

Anand Jain
Alan G. Jensen
Piotr Karocki
Mark J. Knight
John Knowles
Yeou Song Lee
Adam W. Ley
Teresa P. Lopes
G. L. Luri
Mark F. Mcgranaghan
Gary L. Michel
Scott Misha
William J. Mitchell
Mukund U. Modi
Ion A. Neag
Jay J. Nemeth-Johannes
Michael S. Newman
Charles Kamithi Negethe
Leslie A. Orlidge

Stephen A. Osella
Donald M. Parker
Ulrich Pohl
Vikram Punj
John E. Ralph
Gordon D. Robinson
Robert A. Robinson
David A. Rohacek
Randall M. Safier
Bartien Sayogo
Thomas Schossig
Mike Seavey
John W. Sheppard
Roger J. Sowada
Josph J. Stanco
Walter Struppler
Ronald G. Taylor
Vincent J. Tume
Srinivasa R. Vemuru
Oren Yuen

When the IEEE-SA Standards Board approved this standard on 15 September 2006, it had the following membership:

Steve M. Mills, *Chair*
Richard H. Hulett, *Vice Chair*
Don Wright, *Past Chair*
Judith Gorman, *Secretary*

Mark D. Bowman
Dennis B. Brophy
Joseph Bruder
Richard Cox
Bob Davis
Julian Forster*
Joanna N. Guenin
Mark S. Halpin
Raymond Hapeman

William B. Hopf
Lowell G. Johnson
Herman Koch
Joseph L. Koepfinger*
David J. Law
Daleep C. Mohla
Paul Nikolich

T. W. Olsen
Glenn Parsons
Ronald C. Petersen
Gary S. Robinson
Frank Stone
Malcolm V. Thaden
Richard L. Townsend
Joe D. Watson
Howard L. Wolfman

*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Satish K. Aggarwal, *NRC Representative*
Richard DeBlasio, *DOE Representative*
Alan H. Cookson, *NIST Representative*

Don Messina
IEEE Standards Project Editor

William Ash
IEEE Standards Program Manager, Technical Program Development

Contents

1. Overview	1
1.1 Scope	1
1.2 Purpose	1
1.3 Application	2
1.4 Conventions used within this document	2
2. Normative references.....	3
3. Definitions, acronyms, and abbreviations	3
3.1 Definitions	3
3.2 Acronyms and abbreviations	5
4. Automatic Test Markup Language	6
4.1 Background.....	6
4.2 Purpose	6
4.3 ATML and the product life cycle	6
4.4 ATML framework	9
4.5 Specification techniques	11
4.6 ATML component standards	13
4.7 XML schema names and locations	15
5. Conformance	16
6. Extensibility.....	16
Annex A (normative) XML schema style guidelines	17
Annex B (normative) ATML common element schemas.....	25
Annex C (informative) Architecture examples.....	78
Annex D (informative) Glossary	84
Annex E (informative) Bibliography.....	87

IEEE Standard for Automatic Test Markup Language (ATML) for Exchanging Automatic Test Equipment and Test Information via XML

IMPORTANT NOTICE: *This standard is not intended to assure safety, security, health, or environmental protection in all circumstances. Implementers of the standard are responsible for determining appropriate safety, security, environmental, and health practices or regulatory requirements.*

This IEEE document is made available for use subject to important notices and legal disclaimers. These notices and disclaimers appear in all publications containing this document and may be found under the heading “Important Notice” or “Important Notices and Disclaimers Concerning IEEE Documents.” They can also be obtained on request from IEEE or viewed at <http://standards.ieee.org/IPR/disclaimers.html>.

1. Overview

The family of Automatic Test Markup Language (ATML) standards is being developed under the guidance of the Test Information Integration (TII) subcommittee of the IEEE Standards Coordinating Committee 20 (SCC20) to serve as standards for product test. The ATML family of standards specifies a comprehensive environment for integrating design data, test strategies and requirements, test procedures, test results management, and test system implementations. The family of ATML standards includes reference to IEEE Std 1232™ (AI-ESTATE), IEEE P1636™ (SIMICA), IEEE Std 1522™ (Testability and Diagnosability Characteristics and Metrics), and IEEE Std 1641™ (STD). These referenced IEEE standards are therefore part of the ATML family.¹

1.1 Scope

ATML defines a standard exchange medium for sharing information between components of automatic test systems. This information includes test data, resource data, diagnostic data, and historic data. The exchange medium is defined using the eXtensible Markup Language (XML). This document specifies the framework for the family of ATML standards.

1.2 Purpose

The purpose of ATML is to support test program, test asset, and Unit Under Test (UUT) interoperability within an automatic test environment. ATML accomplishes this through a standard medium for exchanging UUT, test, and diagnostic information between components of the test system. The purpose of this document is to provide an overview of ATML goals as well as to provide guidance for usage of the ATML family of standards.

¹ Information on references can be found in Clause 2.

1.3 Application

This standard provides an overview of the ATML family of standards for developing the following:

- a) ATML-conformant systems.
- b) Design data for use in test.
- c) ATML environment for tools.
- d) Shared usage of maintenance data and the results of testing.

1.3.1 Users

Anticipated users of the ATML family of standards include the following:

- a) UUT developers.
- b) UUT maintainers.
- c) Test Program Set (TPS) developers.
- d) TPS maintainers.
- e) Automatic Test Equipment (ATE) system developers.
- f) ATE system maintainers.
- g) Test instrument developers.
- h) Developers of ATML-based tools and systems.
- i) Developers of prime mission equipment that use the supported UUT as a component.

1.4 Conventions used within this document

This standard presents an overview (background and purpose) of ATML, how ATML applies to the life cycle of a product, and defines the ATML framework.

This standard uses the vocabulary and definitions of relevant IEEE standards. In case of conflict of definitions, except for those portions quoted from standards, the following precedence shall be observed: (1) Clause 3, (2) Annex E, and (3) *The Authoritative Dictionary of IEEE Standards Terms* [B3].²

For clarity, portions of IEEE Std 1232, IEEE Std 1522, IEEE P1636, and IEEE Std 1641 have been repeated within this standard. In the event of revision to these standards, the current, approved version of that IEEE standard takes precedence.

² The numbers in brackets correspond to those of the bibliography in Annex E.

2. Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

IEEE P1636™ (Draft 0.2), IEEE Standard for Software Interface to Maintenance Information and Collection Analysis (SIMICA).^{3, 4, 5}

IEEE Std 1232™ -2002, IEEE Standard for Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE).

IEEE Std 1522™ -2004, IEEE Trial-Use Standard for Testability and Diagnosability Characteristics and Metrics.

IEEE Std 1641™ -2004, IEEE Standard for Signal and Test Definition (STD).

3. Definitions, acronyms, and abbreviations

3.1 Definitions

For the purposes of this standard, the following terms and definitions apply. *The Authoritative Dictionary of IEEE Standards Terms* [B3] should be referenced for terms not defined in this clause. In the event a term is explicitly redefined, or further defined in an ATML component standard, the component standards definition shall be normative for that ATML component standard.

3.1.1 application: (A) The use to which a system is put. **(B)** The use of capabilities provided for by a system specific to the satisfaction of a set of users' requirements.

3.1.2 ATML instance document: *See: instance document.*

3.1.3 ATML namespace: A collection of names, identified by a URI reference. (adapted from "Namespaces in XML" [B18])

3.1.4 diagnostic data: That information supporting the investigation and analysis of the cause or nature of a condition, situation, or problem through all phases of a system life cycle.

3.1.5 element: A bounded component of the logical structure of an XML document that has a type and that may have XML attributes and content. (adapted from "XML 1.0 Recommendation"[B2])

3.1.6 entity: A distinct thing, object, or concept. (adapted from *The Artificial Intelligence Dictionary* [B20])

3.1.7 fault: A degradation in performance due to detuning, mis-adjustment, mis-alignment, or failure of part(s).

³ IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, Piscataway, NJ 08855, USA (<http://standards.ieee.org/>).

⁴ This IEEE standards project was not approved by the IEEE-SA Standards Board at the time this publication went to press. For information about obtaining a draft, contact the IEEE.

⁵ The IEEE standards or products referred to in this clause are trademarks of the Institute of Electrical and Electronics Engineers, Inc.

3.1.8 framework: A framework is a real or conceptual structure intended to serve as guidance for the building of something that expands this structure into something useful.

3.1.9 functional test: A test that is intended to verify that a test subject is behaving as specified.

3.1.10 global attribute: An attribute declaration that is a child of the `xs:schema` element. A global attribute can be applied to any element.

3.1.11 instance document: An information set, grouped for some purpose, which is governed by a single XML Schema.

3.1.12 integrated diagnostics: A structured process that maximizes the effectiveness of diagnostics by integrating testability, automatic and manual testing, historical data, training, and technical information to provide a cost-effective capability to detect and isolate faults with the ultimate goal of maximizing equipment availability and minimizing total user cost.

3.1.13 language independent specification: The format for descriptions not tied to any specific computer language.

3.1.14 manufacturing defect: (A) A product anomaly. (B) Any nonconformance with the specified requirements of the product.

3.1.15 object: A member (instance) of a class that encapsulates the data (state) and the behavior (methods) of the object. (adapted from *The Artificial Intelligence Dictionary* [B20])

3.1.16 object instance: A specific occurrence of an object.

3.1.17 signal: (A) The behavior controlled or observed by a test resource. (B) A visual, audible, or other indication used to convey information.

3.1.18 static parameters: Variables given a constant value for a specific purpose or process.

3.1.19 stimulus: Any physical or electrical input applied to a test subject intended to produce a measurable response. (adapted from MIL-STD-1309D [B23])

3.1.20 testability: A design characteristic that allows the status (operable, inoperable, or degraded) of an item to be determined and the isolation of faults within the item to be performed in a timely and efficient manner. (adapted from MIL-STD-1309D [B23])

3.1.21 test entity: A specific procedure or action that will be taken to determine a test subject's capabilities or limitations.

3.1.22 unicode: An industry standard designed to allow text and symbols from all languages to be consistently represented and manipulated.

3.1.23 Unicode Transformation Format (UTF): A variable-length character encoding for unicode.

3.1.24 XML schema: A description of a type of XML document, typically expressed in terms of constraints on the structure and content of documents of that type, above and beyond the basic syntax constraints imposed by XML itself. An XML schema provides a view of the document type at a relatively high level of abstraction.

3.2 Acronyms and abbreviations

ABBET	A Broad Based Environment for Test
AI-ESTATE	Artificial Intelligence Exchange and Service Tie to All Test Environments
API	Application Programming Interface
ARINC	Aeronautical Radio Incorporated
ATE	Automatic Test Equipment
ATLAS	Abbreviated Test Language for All Systems
ATML	Automatic Test Markup Language
ATPG	Automatic Test Program Generation
ATS	Automatic Test System
BIT	Built-In Test
CAD	Computer-Aided Design
CAGE	Commercial and Government Entity
CORBA	Common Object Request Broker Architecture
COM	Component Object Module
DLL	Dynamic Link Library
DMC	Diagnostics and Maintenance Control
HTML	HyperText Markup Language
ID	Identifier
ITA	Interface Test Adapter
I-Level	Intermediate Level
IVI®	Interchangeable Virtual Instrumentation
LAN	Local Area Network
LRU	Line Replaceable Unit
O-Level	Operational Level
RFI	Receiver/Fixture Interface
SCC20	Standards Coordinating Committee 20
SCPI	Standard Commands for Programmable Instrumentation
SIMICA	Software Interface for Maintenance Information Collection and Analysis
SRU	Shop Replaceable Unit
STD	Signal and Test Definition
TAD	Test and ATS Description
TEDL	Test Equipment Description Language
TII	Test Information Integration
TP	Test Program
TPS	Test Program Set
URI	Uniform Resource Identifier
URN	Uniform Resource Name
URL	Universal Resource Locator
UTC	Coordinated Universal Time
UTF-8	8-bit Unicode Transformation Format
UUT	Unit Under Test
VME	Versa Module Europa
VPP	VXI Plug & Play
VXI	VMEbus Extensions for Instrumentation
W3C	World Wide Web Consortium
WSDL	Web Services Definition Language
XHTML	eXtensible HyperText Markup Language
XML	eXtensible Markup Language
XSD	XML Schema Document

4. Automatic Test Markup Language

4.1 Background

Market and technology pressures cause the need for improved testing environments. There are market pressures to reduce a new product's time-to-market and to reduce maintenance costs. At the same time, testing environments must respond to continuously evolving and increasingly complex technology used in modern products. Test technology needs to keep pace with these pressures.

The lack of accepted industry standards for test related information is one factor that has limited the availability and use of improved test environments. Computer-aided design (CAD) environments and exchange standards for design data have helped designers to cope with these pressures, but testing environments have lagged behind.

Computer-aided test environments offer the potential to efficiently access and manipulate design and test information. For example, product test information from earlier life-cycle phases can be preserved to eliminate rediscovery efforts in later phases. Libraries of reusable test procedures can be provided for various types of products. Test equipment interfaces, capabilities, and control characteristics can also be accessed from libraries and used to adapt existing tests to other environments with different test resources.

4.2 Purpose

The primary purpose of this standard shall be to specify a family of standards for test environments that encompass the total product life cycle. The family of standards shall define an integrated set of test-related information that supports the information needs of test environments for testing applications. This family of standards should accomplish the following objectives:

- a) Facilitate the communication, sharing, and reuse of product design and test information for the purpose of testing the product.
- b) Facilitate TPS portability and interoperability.
- c) Facilitate instrument interchangeability.
- d) Facilitate the development, integration, and use of test software and test software development tools.
- e) Support the application of integrated diagnostics.
- f) Support modular software architectures based on a framework that supports reusable software products.

4.3 ATML and the product life cycle

There are primarily five reasons for performing tests on products:

- a) *Design verification.* Verify that a product design meets its functional performance specifications.
- b) *Product verification.* Verify that a product meets its functional and structural specifications.
- c) *Product calibration.* Adjust the characteristics of a product to meet specified tolerances or performance criteria. Calibration tests are often incorporated as part of a verification process.

- d) *Product maintenance*. Identification of a fault condition(s) in the event a product verification test fails or a problem is reported with the product.
- e) *Process control*. Acquire information about a process used to produce, operate, or maintain a product. For example, monitor and maintain proper environmental conditions during a manufacturing process.

These reasons motivate the creation of test assets. Test assets can be manually or automatically controlled, extended from simple inspection to complex functional processing, be built in or applied by external means, or be applied on entire systems or constitute subassemblies. The decisions regarding the test assets depend on the selected life-cycle strategies. All tests produce potentially useful data that can be used for evaluation as dictated by either technology or policy. ATML supports the useful exchange of test information among product life-cycle phases while supporting the different test-related technologies present in each.

4.3.1 Product life cycle

The product life cycle shown in the top portion of Figure 1 is typical of product development, deployment, operation, and maintenance operations. The life-cycle phases are shown from left to right across the diagram. Activities performed during each phase are shown as circles on the diagram, with the test activities being highlighted. The three life-cycle phases that have test activities are as follows:

- a) *Design*. The product proceeds from a concept to a working prototype or preproduction model. The characteristics of the model are tested to verify compliance with specified design and performance criteria.
- b) *Manufacture*. The product is fabricated and made available for distribution and use. Tests are used to assure that the product is properly assembled prior to delivery.
- c) *Support*. The product is performing in its intended function. Tests are used to monitor the products condition during normal operation, to support periodic maintenance activities, and to diagnose faults in trouble situations.

Figure 1 also indicates that products frequently have multiple levels of assembly (e.g., circuit card, subsystem, and system) and that test activities occur at all levels within each life-cycle phase.

4.3.2 Test information

The lower portion of Figure 1 identifies five categories of test information associated with every test application, as follows:

- a) *Design data*. This includes characteristics of the test subject that must be known before tests can be defined. It can include both static structural properties and dynamic behavior.
- b) *Test strategy*. This includes the targeted set of test subject faults, specifications for a set of individual tests, and diagnostic knowledge. The targeted set of test faults is the set of defects that the test application is expected to detect if present. Each test is characterized by a measured test subject characteristic, its acceptable range of values, and the method used for making the measurement. Diagnostic knowledge correlates test outcomes with the test subject faults that they reveal. The test strategy within a specific test context contributes to the test requirements that must be defined and met.
- c) *Test control*. This consists of manual or automatic test procedures for the test subject.
- d) *Test resources*. This includes descriptions of test equipment configurations, capabilities, and control characteristics.

- e) *Test and maintenance information.* This consists of historical data collected during performance of tests. It can include measured values for each test, recommended corrective actions, and maintenance actions that were performed. This collected information can be used within and between various phases of the product life cycle to assure proper maturation of the testability and diagnostics of a product, as well as the product itself.

The lower portion of Figure 1 also shows several paths where improved flow of test information can reduce the overall cost of test. The solid arrows represent information flow dependencies during the development process and the subsequent feedback of actual test results within a single test application. However, test information is rarely in a form that other, related test applications can readily access and use (e.g., by integrated diagnostics). The dotted arrows indicate that frequently the transfer of test information from one level of system integration to another is nearly nonexistent.

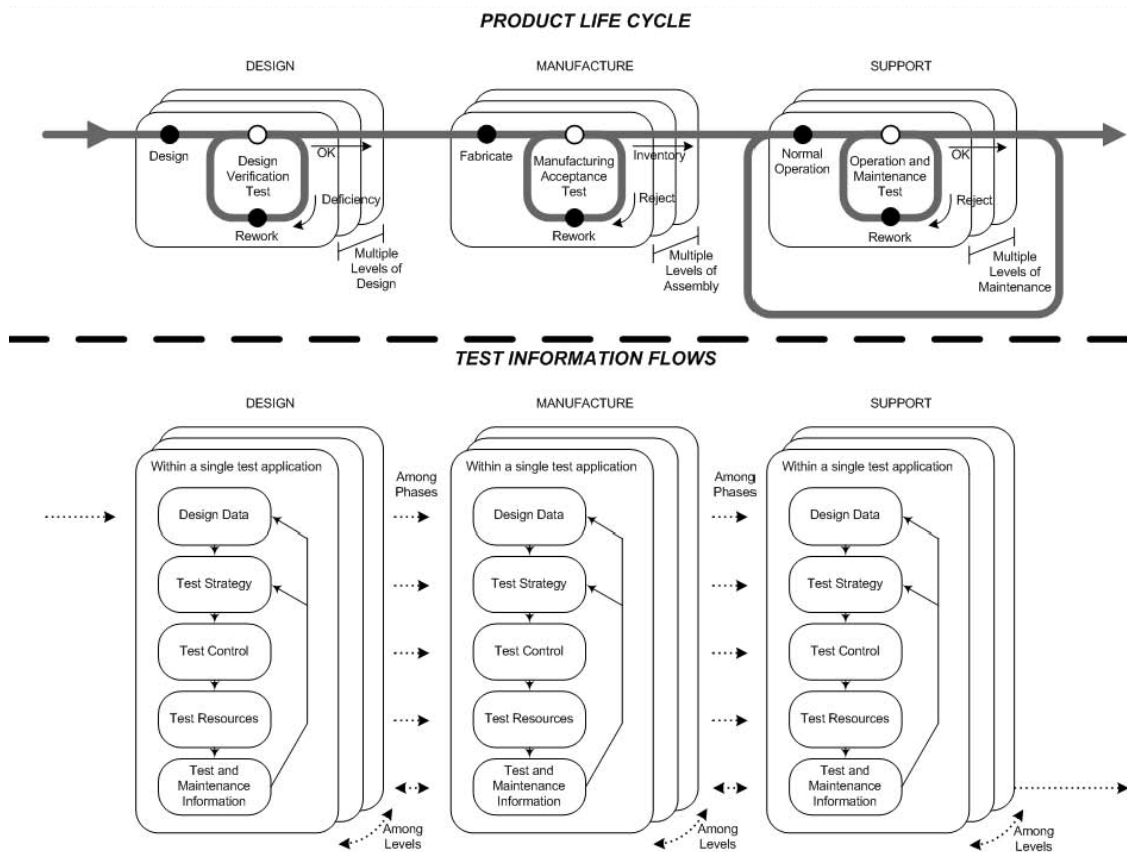


Figure 1—Product life cycle and test information flow

The use of different test systems and proprietary data formats has inhibited the exchange of useful test information. Defining a family of standards for representing test information promotes the exchange among diverse test environments. In addition, these standards promote the feedback of test and maintenance information, enabling continuous improvement of both the product and its test processes. This is accomplished by strengthening the interfaces between activities and phases of the product life cycle, thus promoting free flow of required information. Results of experience supporting a product can be used to recommend changes in product design, which would be fed forward to the design phase for the next generation of the product (indicated by the start of a new product life cycle).

It may not always be practical to completely reuse test information in subsequent life-cycle phases. The amount of reuse that can be achieved depends on the context within which the testing occurs, the extent to which common test information applies, and the availability of test resources with the required capabilities.

Recognizing these limitations, these standards organize test information in a way that allows the useful elements to be found and reapplied. The effort needed to capture a sufficiently rich set of test information is an investment that is recouped when the pertinent information is found and reused in other test applications.

4.4 ATML framework

In general, a framework is a real or conceptual structure intended to serve as guidance for the building of something that expands this structure into something useful.

In the context of ATML, the structure shall be the component standards, the useful structure should be the automatic test system (ATS), and the guidance to build the useful structure shall be this standard.

The ATML framework has been developed to

- a) Summarize and organize the essential elements of an ATS.
- b) Provide a common frame of reference.
- c) Eliminate the need to use a variety of custom file formats.
- d) Provide compliance with the World Wide Web Consortium (W3C) standards.
- e) Be based on standards.
- f) Be extensible.
- g) Enable the creation of modular ATS architectures (components based on the ATML component standards can easily be substituted and data can be shared between the components).

The ATML component standards shall be the core elements of the ATML framework. Figure 2 portrays the integrated family of standards that make up the ATML framework, with example applications.

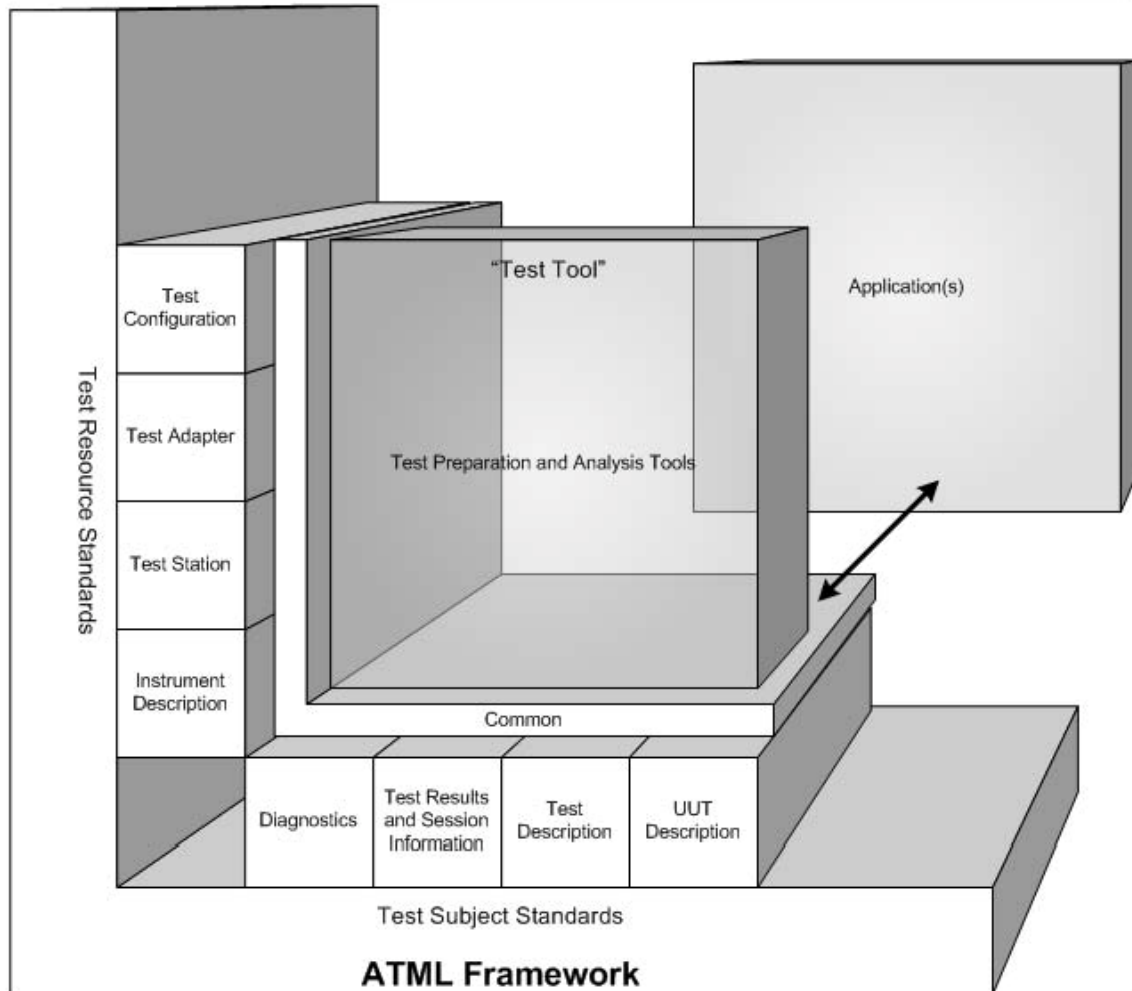


Figure 2—ATML framework, components, and compliant applications

The test subject standards, represented by the horizontal axis of Figure 2, shall support the capture and reuse of test subject information. Opportunities for reuse of test subject information include situations where products are tested in various phases of the product life cycle and situations where common components are used in multiple products. Test subject information captures specifications for test subject design and test requirements, which avoids rediscovery efforts in the initial development, maintenance, and rehosts to test applications. The test subject information also includes diagnostic knowledge that can be accessed during the test process, such as recommended corrective actions, based on the test results.

Test resource standards, represented by the vertical axis of Figure 2, apply to test resource control and information. Test resource standards shall support specifications for test application resource requirements and test equipment capabilities. These standards support adaptation of test applications to changes in test equipment configurations, which can range from the replacement of a single instrument to rehost in a separate test environment.

As stated, the ATML framework is defined in the form of ATML components. The ATML components define the domain-orientated information. ATML components are segmented into

- a) ATML component standards (in the form of formal IEEE standards).
- b) XML schemas.

- c) Reference to ATML instance documents. (However, specific instance documents shall not be part of the ATML framework.)

4.4.1 ATML component standards

The ATML components (as denoted in 4.6) shall have an associated IEEE published standard. Each of the ATML component standards shall contain the definition, description, and use of each element of the ATML component.

4.4.2 XML schemas

The ATML components shall have an associated XML schema (as denoted in 4.7). XML schemas are described in 4.5.1.

XML schemas shall be located on the World Wide Web at the locations specified in 4.7.

4.4.3 ATML instance documents

ATML instance documents are a collection of specific information defined and organized by the referenced XML schema. (e.g., a particular instrument's instance document shall contain the definition of the particular instrument, per the instrument description XML schema specification)

4.5 Specification techniques

The use of XML to define the ATML framework is discussed in the following sub-clauses.

4.5.1 XML schemas and their use in ATML

The XML schema is the formal language used to specify information requirements. The language focuses on the definition of entities, which are the objects of interest. The entities are defined in terms of their elements and attributes, which are the traits or characteristics considered to be important for using and understanding them. These elements or attributes have a representation, which might be a simple data type (such as integer) or another entity type. The XML schema also specifies constraints, rules, and relationships between entities.

ATML uses XML schemas to precisely specify the data that can reside in an ATML test environment. XML schemas shall be specified for those categories of test information where different sets of data can be instantiated and exchanged between ATML implementations. Test information that conforms to the ATML schemas should be accessed and manipulated by software tools in an ATML test environment. The guidelines for the development of ATML XML schemas are provided in Annex A.

The advantage of XML schemas is that they provide for a uniform representation of data that is in a structured format, which can be used across applications, is open for extensions, and represented in an idealized format (not just a “dump of data”).

It should be noted that some constraints cannot be represented in the XML schemas, and consequently are specified only in the corresponding standards. Thus, the sources for the complete set of requirements are the standards and the schemas. Moreover, validation of instance documents against XML schemas does not guarantee that the instance documents satisfy completely the ATML requirements; additional verification may be necessary, for constraints that are not expressed in schemas.

4.5.2 Relationships between ATML and the IEEE AI-ESTATE, SIMICA, testability and diagnosability characteristics and metrics, and STD standards

Certain ATML component standards (and their associated XML schemas) utilize or reference capabilities defined in existing IEEE standards and projects under development. Specifically, ATML component standards utilize or reference the following:

- a) IEEE Std 1232 (Standard for Artificial Intelligence Exchange and Service Tie to All Test Environments): This standard provides the means to exchange and process diagnostic information, as well as control the diagnostic processes (processes such as: diagnostic reasoning, diagnosability assessment or maintenance support).
- b) IEEE Std 1522 (Trial-Use Standard for Testability and Diagnosability Characteristics and Metrics). This standard defines technology-independent testability and diagnosability characteristics and metrics, particularly those based on relevant standard information models, including standard AI-ESTATE information models.
- c) IEEE P1636 (Standard for Software Interface to Maintenance Information and Collection Analysis): This draft standard provides the means to store and retrieve machine processable representations of historical diagnostic and maintenance information. (processes such as testability analysis, or diagnostic maturation)
- d) IEEE Std 1641 (Standard for Signal and Test Definition): This standard provides the means to define and describe signals used in testing.

4.5.2.1 IEEE Std 1232 (AI-ESTATE)

ATML Diagnostics requires a means to exchange and process diagnostic information, as well as control the diagnostic process. All of these ATML Diagnostic requirements will be supported by the utilization of the AI-ESTATE standard. Should any ATML Diagnostic requirement not be satisfied by the AI-ESTATE standard, the requirement(s) shall be brought to the IEEE SCC20 DMC subcommittee in the form of a change proposal to the AI-ESTATE standard (ATML Diagnostics should not become a “competing standard” to AI-ESTATE).

4.5.2.2 IEEE Std 1522 (Testability and Diagnosability Characteristics and Metrics)

ATML Diagnostics requires a consistent, unambiguous definition of testability and diagnosability characteristics and metrics. All of these ATML Diagnostic requirements will be supported by the utilization of the Testability and Diagnosability Characteristics and Metrics standard. Should any ATML Diagnostic requirement not be satisfied by the Standard, the requirement(s) shall be brought to the IEEE SCC20 Diagnostics and Maintenance Control (DMC) subcommittee in the form of a change proposal to the IEEE 1522 standard (ATML Diagnostics should not become a “competing standard” to IEEE Std 1522).

4.5.2.3 IEEE P1636 (SIMICA)

ATML Diagnostics requires a means to store and retrieve historical diagnostic and maintenance information. All of these ATML Diagnostic requirements will be supported by the utilization of the SIMICA project. Should any ATML Diagnostic requirement not be satisfied by the SIMICA project, the requirement(s) shall be brought to the IEEE SCC20 DMC subcommittee in the form of a change proposal to the SIMICA project (ATML Diagnostics should not become a “competing standard” to SIMICA).

4.5.2.4 IEEE Std 1641 (STD)

ATML Test Description, Instrument Description, Test Adapter, and Test Station all utilize the concept of a signal description within the Standards specification and within the XML schema. Where this occurs, the IEEE Std 1641 construct(s) will be referenced. ATML component standards do not redefine, repeat, or

compete with the constructs defined in IEEE Std 1641. Should any ATML signal description requirement not be satisfied by the STD standard, the requirement(s) shall be brought to the IEEE SCC20 Test and ATS Description (TAD) subcommittee in the form of a change proposal to the STD standard.

4.6 ATML component standards

The family of ATML standards (along with its associated XML schemas) defines a logically related set of ATML information. These ATML component standards elaborate on information that only appears as “place holders” in this standard. The ATML family of standards is defined in Table 1.

Table 1—ATML components

Component name	Brief description	Standard
Standard Automatic Test Markup Language (ATML) for Exchanging Automatic Test Equipment and Test Information via XML	This standard	IEEE Std 1671™ (ATML)
Common	Annex B of this standard. Contains the shared type definitions utilized within two or more ATML components	IEEE Std 1671™ (ATML)
Diagnostics	Supports the execution and analysis of diagnosis and diagnostic procedures	IEEE Std 1232™ (AI-ESTATE) IEEE Std 1522™ IEEE P1636™ (SIMICA) IEEE P1636.2™ [B8] (SIMICA: Maintenance Action Information)
Instrument Description	Provides for the description of an test instrument	IEEE P1671.2™ [B10] (ATML: Instrument Description)
Test Adapter	Provides for the description of an Interface Test Adapter (ITA)	IEEE P1671.5™ [B13] (ATML: Test Adapter)
Test Configuration	Provides for the description of the testing configuration	IEEE P1671.4™ [B12] (ATML: Test Configuration)
Test Description	Provides for the description of the test subjects test requirements	IEEE P1671.1™ [B9] (ATML: Test Description)
Test Results and Session Information	Contains the results of a single run of an test, or tests, performed on a test subject	IEEE P1636.1™ [B7] (SIMICA: Test Results and Session Information)
Test Station	Provides for the description of a test station	IEEE P1671.6™ [B14] (ATML: Test Station)
UUT Description	Provides for the description of a test subject	IEEE P1671.3™ [B11] (ATML: UUT Description)

4.6.1 Common

Common shall provide the definition of common types and attributes that are used by more than one of the XML schemas. Common, as a result, shall simply be a “toolbox” for other XML schemas.

ATML defines two common schemas: Common.xsd and HardwareCommon.xsd.

Common.xsd provides ATML unique types and attributes, HardwareCommon.xsd provides ATML hardware unique types and attributes, as well as includes the Common.xsd schema.

Common is defined in Annex B.

4.6.2 Diagnostics

Diagnostics facilitates diagnostic information sharing to support the execution and analysis of diagnosis and diagnostic procedures.

While ATML Diagnostics is strongly focused around artificial intelligence type applications, the ATML definition of diagnostic data is broad enough to not preclude some other maintenance related standards from being developed at some future date.

The process of diagnosis will include information from potentially all of the ATML components. The specific diagnostic information for the Diagnostics component is found in IEEE Std 1232™, IEEE Std 1522™, IEEE P1636™, and P1636.2™ [B8].

4.6.3 Instrument description

Instrument Description facilitates the definition of the static description of an Instrument. The Instrument Description will facilitate the descriptions of a single instrument, or of synthetic / virtual / composite instrumentation.

4.6.4 Test adapter

Test Adapter facilitates the unique description of the interface between the UUT and the Test Station, the physical and electrical characteristics, the capabilities / performance, the identification and classification, etc. This includes the cables, connectors, wires, contacts, etc.

4.6.5 Test configuration

Test Configuration facilitates the identification of all the hardware, software and documentation necessary to test a UUT on a particular ATS.

4.6.6 Test description

Test Description facilitates the definition of the test performance, test conditions, diagnostic requirements, and support equipment to locate faults, align, and verify proper operation of a UUT.

Signal descriptions within a Test Description can be represented utilizing IEEE Std 1641™ (STD).

Test Description(s) will be utilized in the development of a TPS.

4.6.7 Test results and session information

Test Results and Session Information provides the definition for the data collected that resulted from executing test(s) of a UUT in manual, semi-automatic or fully automatic test environments. This includes the measured values, pass/fail results, and accompanying data including test operator, station information, environmental conditions, etc.

IEEE P1636.1™ [B7] is a component standard of IEEE P1636™.

4.6.8 Test station

Test Station facilitates the specification of a particular automatic test station. This includes the physical and electrical characteristics, the paths between test system ports and the Instrument's, tolerances and accuracy of the test station, test station identification information such as part number, serial number, nomenclature,

location; status information such as calibration data, dates, and self test status; operational history, such as system up-time, external interfaces, safety information such as interlocks, temperature sensing; power requirements controller definitions, etc.

4.6.9 UUT Description

UUT Description facilitates the unique description of a particular UUT. This includes information such as the name, part number, model number, type, power requirements, interfaces, physical properties, and operational requirements.

4.7 XML schema names and locations

The ATML family of standards, their associated XML schemas names, and where each of the XML schemas shall be located, are as defined in Table 2.⁶

Table 2—XML schema names and locations

Component standard	XML schema	XML schema location
ATML (This Standard)	None	None
Diagnostics	BayesModel.xsd	To be determined by the IEEE
	CommonElementModel.xsd	To be determined by the IEEE
	DynamicContextModel.xsd	To be determined by the IEEE
	DynamicInferenceModel.xsd	To be determined by the IEEE
	EnhancedDiagnosticInferenceModel.xsd	To be determined by the IEEE
	FaultTreeModel.xsd	To be determined by the IEEE
	maid.xsd	To be determined by the IEEE
Instrument Description	InstrumentDescription.xsd	To be determined by the IEEE
Signal and Test Definition	stdbsc.xsd stdtsf.xsd	To be determined by the IEEE To be determined by the IEEE
Test Adapter	TestAdapter.xsd	To be determined by the IEEE
Test Configuration	TestConfiguration.xsd	To be determined by the IEEE
Test Description	TestDescription.xsd	To be determined by the IEEE
Test Results and Session Information	TestResults.xsd	To be determined by the IEEE
Test Station	TestStation.xsd	To be determined by the IEEE
UUT Description	UutDescription.xsd	To be determined by the IEEE

⁶ At the time of publication these XSD files were not completed or approved by the IEEE-SA.

Each of the schemas identified in Table 2 may include an ATML Common schema. The ATML Common schema names, and where each of the XML schemas shall be located, are defined in Table 3.

Table 3—ATML common XML schema names and locations

XML schema	XML schema location
Common.xsd	http://standards.ieee.org/downloads/1671/1671-2006
HardwareCommon.xsd	http://standards.ieee.org/downloads/1671/1671-2006

5. Conformance

Conformance to an ATML framework shall be achieved by conforming to one or more ATML component standards. The conformance requirements of the ATML component standard(s) being used shall be satisfied.

6. Extensibility

The provision of an extension mechanism is necessary to ensure the viability of the specification and to allow producers and consumers of ATML instance documents to interoperate in those cases where there is a requirement to exchange relevant data that are not included in the standard XML schema. The use of the extensions shall be done in a way that ensures that a conformant consumer can use the extended file without error, discard or otherwise sidestep the extended data, and use the nonextended portions of the data as it is intended—without error or loss of functionality.

Extensions shall be additional information added to the content model of the element being extended.

Extensions shall not repackaging existing information entities that are already supported by the standard.

An extended instance document shall be accompanied by the extension XML schema and documentation sufficient to explain the need for the extension as well as the underlying semantics and relationship(s) to the base schema.

ATML schemas support two forms of extension, as follows:

- a) Wildcard-based extensions allow for the extension of ATML schemas with additional elements.
- b) Type derivation allows for extending the set of data types by deriving a new type from an existing type.

XML schemas control the location and type of extension allowed. Subclause A.9 describes how to specify the extension points for an XML schema.

Annex A

(normative)

XML schema style guidelines

A.1 Naming conventions

A.1.1 Capitalization conventions

A.1.1.1 Pascal case

The first letter in the identifier and the first letter of each subsequent concatenated word are capitalized.

A.1.1.2 Camel case

The first letter of an identifier is lowercase, and the first letter of each subsequent concatenated word is uppercase.

A.1.1.3 Uppercase

All letters in the identifier are uppercase.

A.1.1.4 Lowercase

All letters in the identifier are lowercase.

A.1.2 Naming guidelines

- a) All words shall be spelled using American English in accordance with the latest edition of *Webster's New Collegiate Dictionary* [B26]. The use of abbreviations and acronyms shall be avoided.
 - 1) As a general rule, acronyms should not be used in XML element and attribute names. When it is necessary to use an acronym, acronyms with three or more characters shall use **Pascal case**. Acronyms with two characters shall use **Uppercase**.
 - 2) Abbreviations shall not be used in XML element and/or attribute names.
 - 3) For XML schema data types, abbreviations shall be avoided while acronyms should not be used.
- b) XML element and XML schema data types shall use **Pascal case**.
- c) Except for XML schema abstract data types, XML schema data type names shall not have the word "Type" appended.
- d) XML attributes should use **Camel case**. There is one exception to this rule. If an element has an "ID" attribute, that attribute should use the **Uppercase** naming convention and be of type **NonBlankString** defined in the ATML Common schema.
- e) Namespace names shall use **Pascal case**.
- f) Namespace prefixes shall use **Lowercase**.

- g) Name segments should be distinguished by the use of mixed case (instead of underscores).
 - 1) Underscores, periods, and dashes shall not be used in XML element, schema data type, or attribute names.
- h) An element that represents a collection shall be named using a plural name.
- i) An element that represents a single collection element (or member) shall be named using a singular name.

A.2 XML declaration

All ATML schema and instance documents shall use an explicit XML declaration as the first line of a file. This declaration shall follow the form: `<?xml version opt._encoding opt._standalone?>`. In general, it is expected that all ATML documents will use UTF-8 encoding and will not use the standalone option. Thus, the XML declaration for ATML documents shall be as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
```

A.3 ATML namespaces

A.3.1 Approved schema namespaces

The namespace URI for approved schemas shall be as follows:

```
http://www.ieee.org/ATML/<release_year>/<schema_name>
```

where

`<release_year>` is the year in which the schema was approved.

`<schema_name>` is the schema name identified in 4.7.

A.3.2 Preapproved schema namespaces

The namespace URI for preapproved (draft, candidate and recommendation) schemas shall be as follows:

```
http://www.ieee.org/ATML/<posting_year>/<version>/<schema_name>
```

where

`<posting_year>` is the year in which the prerelease version of the schema is made available.

`<version>` is an integer that indicates the version of the preapproved schema. The version starts at 01 and increments each time a new preapproved version is made available for evaluation.

`<schema_name>` is the schema name identified in 4.7.

The namespace shall be modified whenever one of the following conditions occurs:

- A change is made to a schema that invalidates existing instance documents (i.e., a major revision update).
- The schema's state changes from "preapproved" to "approved."
- A new "preapproved" version is made available for evaluation.

The use of the schemas is controlled through their namespace, such that any ATML document refers to the namespace when describing one of these components. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<TestStation name="CASS" xmlns="http://www.ieee.org/ATML/2006/TestStation">
  <Configuration>
    <Instrument name="PGEN"
      xmlns="http://www.ieee.org/ATML/2006/InstrumentDescription">
    </Instrument>
    <Instrument name="DMM"
      xmlns="http://www.ieee.org/ATML/2006/InstrumentDescription">
    </Instrument>
  </Configuration>
  <Supports>
    <Equipment>
      <TestAdapter name="123/45"
        xmlns="http://www.ieee.org/ATML/2006/TestAdapter"/>
    </Equipment>
  </Supports>
</TestStation>
```

A.3.3 Target namespace

Every XML schema shall define a target namespace. The namespace shall be defined as a universal resource locator (URL) as described in A.3. Each ATML component XML schema has its own namespace. This provides a standard way to avoid name collisions between schemas.

A.3.4 Default namespace

The default namespace shall be the target namespace.

A.3.5 XML schema namespace reference

The namespace prefix for the XML Schema namespace shall be **xs**:

```
xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

The XML Schema namespace shall not be the default namespace.

A.3.6 Qualified and unqualified

Two attributes of the **xs:schema** element shall be specified for every XML Schema: **elementFormDefault** and **attributeFormDefault**. These attributes specify whether elements and attributes in XML instance documents need to be qualified with the namespace of the XML schema in which they are defined.

The value of **attributeFormDefault** specifies whether attributes in XML instance documents are qualified with the namespace of the XML schema in which they are defined. As an attribute is always defined and used in the context of an element, it is not necessary to qualify the attribute as well as the element.

The value of **attributeFormDefault** shall be **unqualified**.

The value of **elementFormDefault** specifies whether elements in XML instance documents are qualified with the namespace of the XML schema in which they are defined. A value of **qualified** indicates that if the root element is qualified, then all subelements must be qualified as well. A value of **unqualified** indicates that only global elements need to be qualified. Using a value of **unqualified** allows for inconsistent qualification of elements in instance documents.

Given the following example schema with **elementFormDefault** set to **qualified**:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:http://mynamespace.com/MySchema
  xmlns:xs=http://www.w3.org/2001/XMLSchema
  targetNamespace="http://mynamespace.com/MySchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

  <xs:element name="GlobalElement">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ChildElement" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

The following are valid instance documents:

```
<?xml version="1.0" encoding="UTF-8"?>
<GlobalElement xmlns="http://mynamespace.com/MySchema">
  <ChildElement/>
</GlobalElement>
```

and

```
<?xml version="1.0" encoding="UTF-8"?>
<my:GlobalElement xmlns:my="http://mynamespace.com/MySchema">
  <my:ChildElement/>
</my:GlobalElement>
```

The following is not a valid instance document:

```
<?xml version="1.0" encoding="UTF-8"?>
<my:GlobalElement xmlns:my="http://mynamespace.com/MySchema">
  <ChildElement/>
</my:GlobalElement>
```

The value of **elementFormDefault** shall be **qualified**.

A.4 Versioning

The schema version shall be captured in the schema using the version attribute of the schema element.

The format of the schema version shall be <major>.<minor>, where the major portion shall always begin at “0” and the minor portion shall be a two-digit number beginning from “00”.

Previously *released* versions of each schema shall be made available at the URL locations defined in Table 2.

Changes made to an XML schema fall into two categories:

- a) A noninvalidating change—a noninvalidating change is one that does not invalidate existing instance documents. That is, existing instance documents will continue to validate against the new version of the schema. Examples include correcting or adding annotation data, adding an optional element, adding an optional attribute, or adding an enumeration item. For this type of change, it is sufficient to increment the <minor> portion of the version. Although adding optional attributes and elements does not invalidate existing instance documents, new instance documents that take advantage of these new optional elements will not validate against earlier versions of the schema even though the namespace has not changed.